Synthesis of High Level based on Models

Débora Bertasi
Luigi Car (orienting)
bertasi@inf.ufrgs.br

Summary

The use of a high level language for specification of systems has many advantages, including high productivity, reduction of the errors and time of project. This article describes the process to convert functional specifications Java for implementations into the hardware, including the necessary otimizações to the attainment of the biggest amount of possible parallelism. The main profits with this methodology are the reduction in the project time and, the reduction of errors of project due the automatic adaptation of the code to get an optimized ASIC.

1. INTRODUCTION

With the increase of the complexity of the projects of computational systems, a designer became impossible to understand the functionality of one chip or the specification of a system only with diagram of circuits or its logical project. Synthesis of high level is one strong reality for projects of methodologies of systems VLSI.

The automatic synthesis is the transformation of a specification carried through in one determined abstraction level, to a level next to the physical accomplishment, through the application of a set of tools that add structural and/or geometric details to the initial specification [1, 2].

Currently, the automatic synthesis of digital systems from the level of transference between recorders (RT) until the layout level already sufficiently is consolidated, with 2 efficient and commercially available tools razoavelmente [ ]. With this, two strong trends if have detached in the last years: the migration of tools of project for the higher levels of abstraction, notadamente for the algorithmic levels and of system and the adoption of VHDL as standard for the description of circuits.

The adoption of one and another one of these trends in the synthesis of circuits is surrounded of problems and controversies. However, the advantages surpass the problems. Algorithms developed for the synthesis of high level, as allocation and scheduling, come migrando for commercial tools, being based on the great publication number on the 2,3,4,5 subject [ ].

In view of these considerações, a system of automatic synthesis is presented, as boarding of the main tasks of the synthesis of high level. In this direction, a partial system of synthesis was presented in [6], where the tasks compilation of the description of the sequence of operations had been dealt with to be done for an internal representation and posterior scheduling and allocation of resources.

For interpretation, a program developed in the Java language was used and, for the scheduling, the algorithms of synthesis of high level had been implemented, ASAP (soon possible), ALAP (it barks them possible) and List-Scheduling, from a structure of 2 data [ ]. From the existing system this work presents its natural evolution, implementing the generation of the description VHDL, sintetizável for commercial tools as Altera[7] or Mentor[8].

Beyond the some models for the abstraction of code VHDL, this work it supplies to support to specification tool SASHIMI and synthesis of systems based on Java processors, as described in [9], supplying the models the users and specializing the synthesis tools. When, eventually, an application to possess requirements of very demanding time for the processor femtoJava, the Java code will have to be translated to a sintetizável code VHDL, in order that if it obtains to execute the algorithm with bigger speed. The protocol of communication between co-processor VHDL and the Java processor also will be inlaid in the synthesis tool, visa this dedicated being to 9 system SASHIMI [ ].

The description in VHDL generated for the system could be abstracted in some ways, depending it specified model of computation. Each model is specified allowing to capture the behavior of the component of the system governed for a certain particular computational model. As the system already knows which the model of computation chosen for the generation of the ASIC, many referring otimizações to each model could be carried through.

The allowed computational models for the abstraction in the synthesis system will be: Machine of Finite States, Nets of Petri, Pipeline, Dataflow and Aritmético 2.10.1.1 Serial Bit [ ]. It exists a variety of models of computation with express competition and time of different forms. In this work, one adopted the more used models of computation for embarked systems.

2. C@MPARACÃ@ WITH OTHER WORKS

Using the considered system of synthesis, the designer will have first to describe the system using models in Java, this will facilitate the description of the project and will make possible reuses it. After, the designer will go to execute the project in tool SASHIMI verifying the performance of the projected system. The user of the SASHIMI will have to analyze if some routine executes in a bigger time that the waited one.

Although already systems of synthesis of high level exist commercially, the considered system of synthesis is differentiated of other systems of synthesis of high level for supporting some models of computation, and for disponibilizar, in the tool SASHIMI, the models for the designer. That is, when the designer to possess an algorithm that in the synthesis with tool SASHIMI will take much time for the execution, will be able to use the models to speed the process. An identified time one or more routines whose execution does not take care of to the performance requirements, the models will indicate which the best style of synthesis for automatic generation of a ASIC. That is represented in the block Generation of ASICs of figure 1.

In the last years some systems of synthesis of high level had been considered, from there ask because plus synthesis systems it is necessary. A study with some systems of synthesis of high level it was carried through. The analysis of some of them meets to follow it:

In [12] was presented a system of synthesis from a Java description, and the behavior of the generated code always is based on machine of finite states. The described system in [13] presents the abstraction of the computation models machine of finite states and dataflow, leaving of a Java description. In the system presented in [14] uses VHDL as entrance language including mannering specifications. The system supports some levels of description, but the designer must have flexibility to choose the level most appropriate for each part of the project. The exit of the synthesis of high level is a description VHDL in level RT or level of doors that can be synthecized by existing tools.

15 system CALLAS [ ] uses VHDL for both the specification of entrance and the description of the generated hardware. The main characteristics of this system are functional equivalence and temporization of the mannering specification in relation the synthecized structure. The environment includes some transformations of high level and the level-RT in the flow of data and control to optimize area and the delay of the circuit.

It can be noticed that the differential of this work with excessively is in the different models where code VHDL could be abstracted and in the high level of abstraction that the designer can work in its project. As the system already obtains to abstract which the computation model that the user chose, many otimizações could be carried through. In model FSM, if carried through operations to exist being in different states, being able these to be carried through in the same state, that is, not having dependence of data, this otimização could be carried through reducing the number of states. Already in a Dataflow model, the user simply will supply the expression of treatment of the data, without concern of the operations to be carried through with the parallelism maximum, therefore the system will verify the operations to be executed in parallel. Otimizações will be carried through specifically for each model of computation chosen for the user, generating a ASIC with the maximum of pertinent otimizações to each model of computation.

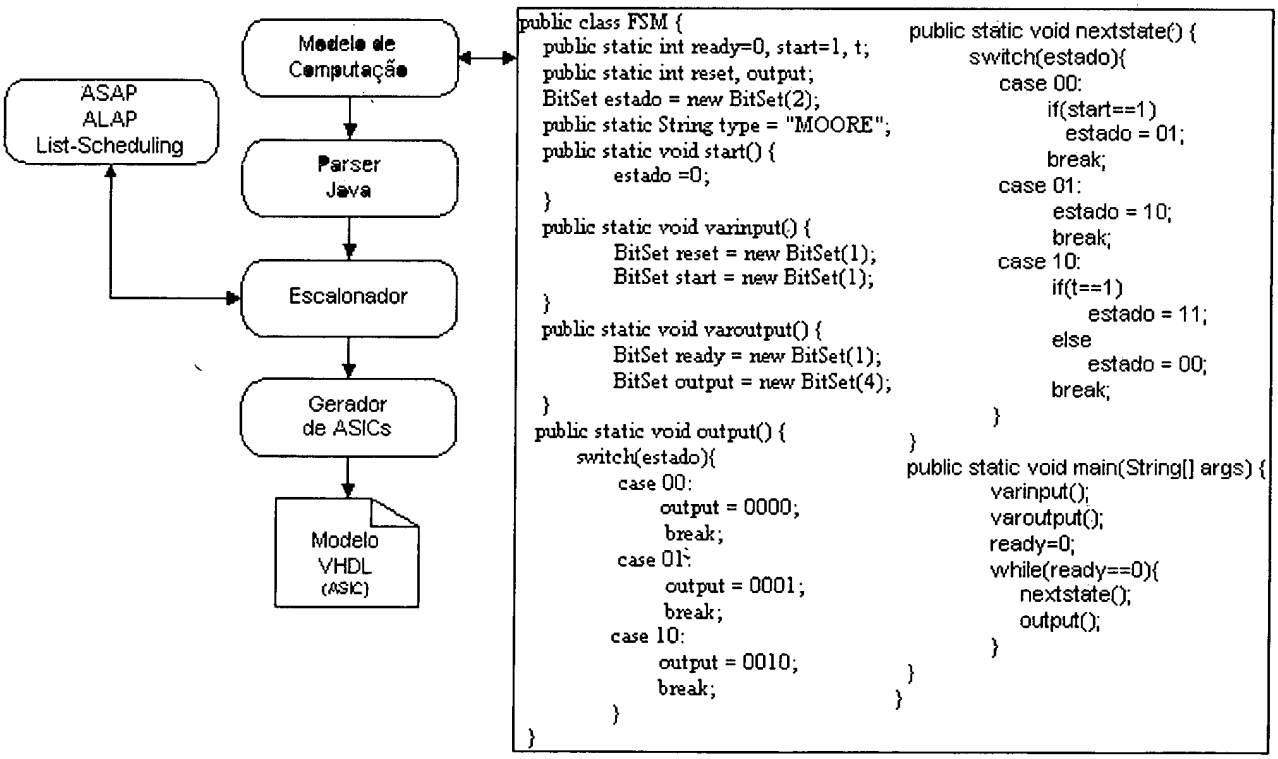

Figure 1? System of Synthesis of High Level and example of classroom FSM

3. REACHED And WAITED RESULTS ALREADY

Until the moment the system if finds in implementation phase. A model for Machines of Finite States was developed. In this model the designer must specify the 0 variable of entrance and exit in different methods. In another method he must specify the corresponding exits to each state. E, finishing, the model must specify the referring operations to each state, as it shows figure 1.

The system uses to parser to interpret the model. Used parser was gotten the homepage of the SUN, generated from 2,7,0 tool 16 ANTLR [ ]. From the information gotten for parser, the structures of data are constructed. The structures contain the abstracted information of the model for posterior generation of code VHDL.

The main difficulties found until the moment had been the search for parser and posterior interpretation of exactly. The interpretation of the code of parser is very excellent in function that the data must be abstracted, of form very needs so that the interpretation of the model is carried through of coherent form to that the designer elaborated.

4. CONCLUSIONS

During the text a boarding for the project of descriptions in Java for generation of a ASIC was presented. In fact, the present work is proving to be possible to implement applications in one high level of abstraction in Java having a ASIC as white device.

The main value added in this project is the abstraction with that the designer can work in its project, since the system will be capable to abstract the model of computation for a description in VHDL. Until the moment a description in VHDL for model FSM already is implemented.

Probably the most important question to be answered would be to determine a minimum frequency, since the system will have to supply one better performance to the application. The implementation of some studies of cases will be useful to help to answer this question.

At the moment, the system is being worked and implemented in Java, becoming the system capable to abstract the some models of computation.

AGRADECIMENTOS

O desenvolvimento desse trabalho só está sendo possível graças ao apoio do CNPq através da concessão de bolsa de mestrado. Ao meu professor orientador Luigi Carro pelas idéias e incentivo, ao amigo Sérgio Akira Ito pela constate ajuda e aos demais amigos do curso de mestrado pela motivação.

REFERÊNCIAS

[1] Wagner, F.R.; Porto, I.J.; Weber,R.F.; Weber, T.S. Métodos de Validação de Sistemas Digitais. Campinas: UNICAMP, 1988.

[2] Gajski, D.; Dutt, N.; Wu, A., Lin,S. High-Level Synthesis: Introduction to Chip and System Design. Kluwer AcademicPublishers. 1992.

[3] Lipset, R. et al. VHDL: Hardware Description and Design. Boston: Kluwer Academic Publishers, 1990.

[4] Glunz, W.; Umbreit, G. VHDL for High-Level Synthesis of Digital Systems. In: European Conference on VHDL, 1990. Proceedings... New York: IEEE, 1990. P.01-11.

[5] Camposano, R. From behavior to structure: high-level synthesis. IEEE Design & test of computers,New York, v.8, n.1, p.43-49, Mar.1991.

[6] Bertasi, D. Implementação parcial de um sistema de síntese de alto nível. Porto Alegre, CPGCC da UFRGS, 2000.

[7] Altera. Data Book. Altera Corporation, San Jose, California, 1996.

[8] Mentor Graphics Corporation. Mentor Graphics System Overview Manual. Mentor Graphics Corporation. Wilsonville, Oregon 1993.

[9] Ito, S. A; Susim, A.; Carro, L.; Jacobi, R. Implementação de uma Máquina Java. In: Quinto Workshop IberChip, 1-3 marco, Peru, 1999, p.252-259.

[10] DI Giacomo, J. VLSI HANDBOOK. New York : Mcgraw-Hill, 1989.

[11] Peres, E. M.; Heuser, C. A. Integrando Aspectos Formais e Informais em uma Linguagem de Anotação de Redes de Petri. Porto Alegre, CPGCC da UFRGS, 1989.

[12] Neto, H.; Cardoso, J.M.P. Macro-Based Hardware Compilation of Java Bytecodes into a Dynamic Reconfigurable Computing System. IEEE Symposium on Field Programmable Custom Computing Machines, Napa Valley, California, USA, April 20-23, 1999.

[13] Girault, A.; Lee, B.; Lee, E. Hierarchical Finite State Machines with Multiple Concurrency Models.IEEE Transactions on Computer-Aided Design, Vol. 18, num 6 june 1999.

[14] Bergamaschi, R.A; Kuehlmann, A. A System for Production Use of High-Level Synthesis. IEEE Transactions on Very Large Scale Integration (VLSI), Vol. 1, No. 3, September 1993.

[15] Biesenack, J.; Koster, M. The Siemens High-Level Synthesis System Callas. IEEE Transactions on Very Large Scale Integration (VLSI), Vol. 1, No. 3, September 1993.

[16] Parr, T.; Lilly, J.; ANTLR Reference Manual.In: http://www.ANTLR.org